Approximating Highly Inapproximable Problems on Graphs of Bounded Twin-Width

Pierre Bergé
Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Édouard Bonnet
Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Hugues Déprés
Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Rémi Watrigant
Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Abstract

For any $\varepsilon > 0$, we give a polynomial-time n^{ε}-approximation algorithm for Max Independent Set in graphs of bounded twin-width given with an $O(1)$-sequence. This result is derived from the following time-approximation trade-off: We establish an $O(1)^{2^q-1}$-approximation algorithm running in time $\exp(O_q(n^{2^{-q}}))$, for every integer $q \geqslant 0$. Guided by the same framework, we obtain similar approximation algorithms for Min Coloring and Max Induced Matching. In general graphs, all these problems are known to be highly inapproximable: for any $\varepsilon > 0$, a polynomial-time $n^{1-\varepsilon}$-approximation for any of them would imply that P=NP [Håstad, FOCS '96; Zuckerman, ToC '07; Chalermsook et al., SODA '13]. We generalize the algorithms for Max Independent Set and Max Induced Matching to the independent (induced) packing of any fixed connected graph H.

In contrast, we show that such approximation guarantees on graphs of bounded twin-width given with an $O(1)$-sequence are very unlikely for Min Independent Dominating Set, and somewhat unlikely for Longest Path and Longest Induced Path. Regarding the existence of better approximation algorithms, there is a (very) light evidence that the obtained approximation factor of n^{ε} for Max Independent Set may be best possible. This is the first in-depth study of the approximability of problems in graphs of bounded twin-width. Prior to this paper, essentially the only such result was a polynomial-time $O(1)$-approximation algorithm for Min Dominating Set [Bonnet et al., ICALP '21].

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Design and analysis of algorithms

Keywords and phrases Approximation algorithms, bounded twin-width

Digital Object Identifier 10.4230/LIPIcs.STACS.2023.10

Related Version *Full Version*: https://arxiv.org/abs/2207.07708

Funding This work was supported by the ANR projects TWIN-WIDTH (ANR-21-CE48-0014) and Digraphs (ANR-19-CE48-0013).

Acknowledgements We thank Colin Geniet, Eunjung Kim, and Stéphan Thomassé for useful discussions.

1 Introduction

Twin-width is a graph parameter introduced by Bonnet, Kim, Thomassé, and Watrigant [7]. Its definition involves the notions of *trigraphs* and of *contraction sequences*. A *trigraph* is a graph with two types of edges: black (regular) edges and red (error) edges. A (vertex) *contraction* consists of merging two (non-necessarily adjacent) vertices, say, u, v into a vertex w,

replacing an edge uz (resp. vz) by wz in the following way: we keep wz black if and only if uz and vz were previously black edges. The other edges incident to w become red (if not already), and the rest of the trigraph remains the same. A *contraction sequence* of an n-vertex[1] graph G is a sequence of trigraphs $G = G_n, \ldots, G_1 = K_1$ such that G_i is obtained from G_{i+1} by performing one contraction. A *d-sequence* is a contraction sequence in which every vertex of every trigraph has at most d red edges incident to it. The *twin-width* of G, denoted by $\mathrm{tww}(G)$, is then the minimum integer d such that G admits a d-sequence. Figure 1 gives an example of a graph with a 2-sequence, i.e., of twin-width at most 2. Twin-width can be naturally extended to matrices (with unordered [7] or ordered [6] row and column sets) over a finite alphabet, and thus to binary structures.

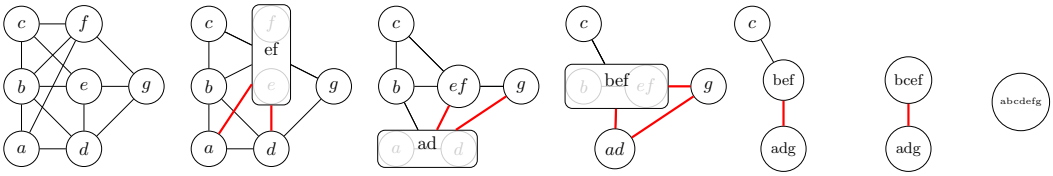

Figure 1 A 2-sequence witnessing that the initial graph has twin-width at most 2.

An equivalent viewpoint that will be somewhat more convenient is to consider a d-sequence as a sequence of partitions $\mathcal{P}_n := \{\{v\} \ : \ v \in V(G)\}, \mathcal{P}_{n-1}, \ldots, \mathcal{P}_1 := \{V(G)\}$ of $V(G)$, such that for every integer $1 \leqslant i \leqslant n-1$, $\mathcal{P}_i$ has i parts and is obtained by merging two parts of $\mathcal{P}_{i+1}$ into one. Now the *red degree* of a part $P \in \mathcal{P}_i$ is the number of other parts $Q \in \mathcal{P}_i$ such that there is in G at least one edge and at least one non-edge between P and Q. A d-sequence is such that no part of no partition of the sequence has red degree more than d. In that case the *maximum red degree* of each partition is at most d. And we similarly get the twin-width of G as the minimum integer d such that G admits a (partition) d-sequence. The *quotient trigraph* $G/\mathcal{P}_i$ is the trigraph G_i, if the (contraction) d-sequence $G_n, \ldots, G_1$ and the (partition) d-sequence $\mathcal{P}_n, \ldots, \mathcal{P}_1$ correspond.

Classes of binary structures with bounded twin-width include graph classes with bounded treewidth, and more generally bounded clique-width, proper minor-closed classes, posets with antichains of bounded size, strict subclasses of permutation graphs, as well as $\Omega(\log n)$-subdivisions of n-vertex graphs [7], and some classes of (bounded-degree) expanders [4]. A notable variety of geometrically defined graph classes have bounded twin-width such as map graphs, bounded-degree string graphs [7], classes with bounded queue number or bounded stack number [4], segment graphs with no $K_{t,t}$ subgraph, visibility graphs of 1.5D terrains without large half-graphs, visibility graphs of simple polygons without large independent sets [3].

For every class $\mathcal{C}$ mentioned so far, $O(1)$-sequences can be computed in polynomial time[2] on members of $\mathcal{C}$. For classes of binary structures including a binary relation interpreted as a linear order on the domain (called *ordered binary structures*), there is a fixed-parameter approximation algorithm for twin-width [6]. More precisely, given a graph G and an integer k, there are computable functions f and g such that one can output an $f(k)$-sequence of G or correctly report that $\mathrm{tww}(G) > k$ in time $g(k)n^{O(1)}$. Such an approximation algorithm

[1] In this introduction, we might implicitly use n to denote the number of vertices, and m, the number of edges of the graph at hand.

[2] Admittedly, for the geometric classes, a representation is (at least partially) needed.

is currently missing for classes of general (not necessarily ordered) binary structures, and in particular for the class of all graphs. We also observe that deciding if the twin-width of a graph is at most 4 is an NP-complete task [2].

We will therefore assume that the input graph is given with a d-sequence, and treat d as a constant (or that the input comes from any of the above-mentioned classes). Thus far, this is the adopted setting when designing faster algorithms on bounded twin-width graphs [7, 5, 26, 23, 15]. From the inception of twin-width [7] –actually already from the seminal work of Guillemot and Marx [17]– it was clear that structures wherein this invariant is bounded may *often* allow the design of parameterized algorithms. More concretely, it was shown [7] that, on graphs G given with a d-sequence, model checking a first-order sentence φ is fixed-parameter tractable –it can be solved in time $f(d, \varphi) \cdot n$–, the special cases of, say, k-Independent Set or k-Dominating Set admit single-exponential parameterized algorithms [5], an effective data structure almost linear in n can support constant-time edge queries [26], the triangles of G can be counted in time $O(d^2 n + m)$ [23].

So far, however, the connection between *having bounded twin-width* and *enjoying enhanced approximation factors* was tenuous. The only such result concerned Min Dominating Set, known to be inapproximable in polynomial-time within factor $(1-o(1)) \ln n$ unless P=NP [12], but yet admits a constant-approximation on graphs of bounded twin-width given with an $O(1)$-sequence [5]. We start filling this gap by designing approximation algorithms on graphs of bounded twin-width given with an $O(1)$-sequence for notably Max Independent Set (MIS, for short), Max Induced Matching, and Coloring. Getting better approximation algorithms for MIS and Coloring in that particular scenario was raised as an open problem [5]. Before we describe our results and elaborate on the developed techniques, let us briefly present the notorious inapproximability of these problems in general graphs.

MIS and Coloring are NP-hard [16], and very inapproximable: for every $\varepsilon > 0$, it is NP-hard to approximate these problems within ratio $n^{1-\varepsilon}$ [19, 27]. The same was shown to hold for Max Induced Matching [9]. Besides, there is only little room to improve over the brute-force algorithm in $2^{O(n)}$: Unless the Exponential Time Hypothesis[3] [21] (ETH) fails, no algorithm can solve MIS in time $2^{o(n)}$ [22] (nor the other two problems). For any r (possibly a function of n) WMIS can be r-approximated in time $2^{O(n/r)}$ [11, 8]. Bansal et al. [1] essentially shaved a $\log^2 r$ factor to the latter exponent. It is known though that polynomial shavings are unlikely. Chalermsook et al. [10] showed that, for any $\varepsilon > 0$ and sufficiently large r (again r can be function of n), an r-approximation for MIS and Max Induced Matching cannot take time $2^{O(n^{1-\varepsilon}/r^{1+\varepsilon})}$, unless the ETH fails. For instance, investing time $2^{O(\sqrt{n})}$, one cannot hope for significantly better than a $\sqrt{n}$-approximation.

Contributions and techniques

Our starting point is a constant-approximation algorithm for MIS running in time $2^{O(\sqrt{n})}$ when presented with an $O(1)$-sequence, which is very unlikely to hold in general graphs by the result of Chalermsook et al. [10].

▶ **Theorem 1.** *On n-vertex graphs given with a d-sequence Max Independent Set can be $O_d(1)$-approximated in time $2^{O_d(\sqrt{n})}$.*

Our algorithm builds upon the functional equivalence between twin-width and the so-called *versatile twin-width* [4]. We defer the reader to Section 2 for a formal definition of versatile twin-width. For our purpose, one only needs to know the following useful consequence of that

[3] That is, the assumption that there is a $\delta > 0$ such that n-variable 3-SAT cannot be solved in time δ^n.

equivalence. From a d'-sequence of G, we can compute in polynomial time another partition sequence $\mathcal{P}_n, \dots, \mathcal{P}_1$ of G of width $d := f(d')$, for some computable function f, such that for every integer $1 \leqslant i \leqslant n$, all the i parts of $\mathcal{P}_i$ have size at most $d \cdot \frac{n}{i}$. Even if some parts of P_i can be very small, this partition is balanced in the sense that no part can be larger than d times the part size in a perfectly balanced partition. Of importance to us is $\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$ when the number of parts ($\lfloor\sqrt{n}\rfloor$) and the size of a larger part in the partition (at most $d\frac{n}{\lfloor\sqrt{n}\rfloor} \approx d\sqrt{n}$) are somewhat level.

We can then properly color the red graph (made by the red edges on the vertex set $\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$) with $d+1$ colors. Any color class X is a subset of parts of $\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$ such that between two parts there are either all edges (black edge) or no edge at all (non-edge). In graph-theoretic terms, the subgraph G_X of G induced by all the vertices of all the parts of X has a simple modular decomposition: a partition of at most $\sqrt{n}$ modules each of size at most $d\sqrt{n}$. It is thus routine to compute a largest independent set of G_X essentially in time exponential in the maximum between the number of modules and the maximum size of a module, that is, in at most $d\sqrt{n}$. As one color class X^* contains more than a $\frac{1}{d+1}$ fraction of the optimum, we get our $d+1$-approximation when computing a largest independent set of G_{X^*}. Figure 2 on page 10 serves as a visual summary of what we described so far.

The next step is to substitute exact exponential algorithms on induced subgraphs of size $O_d(\sqrt{n})$ by recursive calls of our approximation algorithm. Following this inductive process at depth $q = 2, 3, 4, \dots$, we degrade the approximation ratio to $(d+1)^3, (d+1)^7, (d+1)^{15}$, etc. but meanwhile we boost the running time to $2^{O_d(n^{1/4})}, 2^{O_d(n^{1/8})}, 2^{O_d(n^{1/16})}$, etc. In effect we show by induction that:

▶ **Theorem 2.** *On n-vertex graphs given with a d-sequence Max Independent Set has an $O_d(1)^{2^q-1}$-approximation algorithm running in time $2^{O_{d,q}(n^{2^{-q}})}$, for every integer $q \geqslant 0$.*

The following polynomial-time algorithm is a corollary of Theorem 2 choosing $q = O_{d,\varepsilon}(\log\log n)$.

▶ **Theorem 3.** *For every $\varepsilon > 0$, Max Independent Set can be n^ε-approximated in polynomial-time $O_{d,\varepsilon}(1) \cdot \log^{O_d(1)} n \cdot n^{O(1)}$ on n-vertex graphs given with a d-sequence.*

Note that the exponent of the polynomial factor is an absolute constant (not depending on d nor on ε).

We then apply our framework to Coloring and Max Induced Matching.

▶ **Theorem 4.** *For every $\varepsilon > 0$, Coloring and Max Induced Matching admit polynomial-time n^ε-approximation algorithms on n-vertex graphs of bounded twin-width given with an $O(1)$-sequence.*

The main additional difficulty for Coloring is that one cannot satisfactorily solve/approximate that problem on a modular decomposition by simply coloring its modules and its quotient graph. One needs to tackle a more general problem called Set Coloring. Fortunately this generalization is the fixed point we are looking for: approximating Set Coloring can be done in our framework by mere recursive calls (to itself).

For Max Induced Matching, we face a new kind of obstacle. It can be the case that no decent solution is contained in any color class X –in the chosen $d+1$-coloring of the red graph $G/\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$. For instance, it is possible that any such color class X induces in G an edgeless graph, while very large induced matchings exist with endpoints in two distinct color classes. We thus need to also find large induced matchings within the black edges and within the red edges of $G/\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$. This leads to a more intricate strategy intertwining the coloring

of bounded-degree graphs (specifically the red graph and the square of its line graph) and recursive calls to induced subgraphs of G, and to special induced subgraphs of the total graph (i.e., made by both the red and black edges) of $G/\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$.

We then explore the limits of our results and framework in terms of amenable problems. We give the following technical generalization to the approximation algorithms for MIS and Max Induced Matching.

▶ **Theorem 5.** *For every connected graph H and $\varepsilon > 0$, Mutually Induced H-packing admits a polynomial-time n^{ε}-approximation algorithms on n-vertex graphs of bounded twin-width given with an $O(1)$-sequence.*

In this problem, one seeks for a largest induced subgraph that consists of a disjoint union of copies of H. All the previous technical issues are here combined. We try all the possibilities of batching the vertices of H into at most $|V(H)|$ parts of $G/\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$, based on the trigraph that these parts define. For instance with $H = K_2$ (an edge), i.e., the case of Max Induced Matching, the three possible trigraphs are the 1-vertex trigraph, two vertices linked by a red edge, and two vertices linked by a black edge. In the general case, the problem generalization is quite delicate to find. We have to keep some partitions of $V(G)$ and $V(H)$ to enforce that the copies of H in G follow a pattern that the algorithm committed to higher up in the recursion tree, and a weight function on $|V(H)|$-tuples of vertices of G, not to forget how many mutually induced copies of H can be packed *within* these vertices. The other novelty is that some recursive calls are on induced subgraphs of the total graph of $G/\mathcal{P}_{\lfloor\sqrt{n}\rfloor}$ that are *not* induced subgraphs of G. Fortunately, these graphs keep the same bound of versatile twin-width, and thus our framework allows it.

Defining, for a family of graphs $\mathcal{H}$, Mutually Induced $\mathcal{H}$-packing as the same problem where the connected components of the induced subgraph should all be in $\mathcal{H}$, we get a similar approximation factor when $\mathcal{H}$ is a finite set of connected graphs. (Note that Mutually Induced H-packing is sometimes called Independent Induced H-Packing.) In particular, we can similarly approximate Independent H-Packing, which is the same problem but the copies of H need not be induced. (Our approximation algorithms could extend to other H-packing variants without the independence requirement, but these problems can straightforwardly be $O(1)$-approximated in general graphs.)

We can handle some cases when $\mathcal{H}$ is infinite, too. For instance, by slightly adapting the case of MIS, we can get an n^{ε}-approximation when $\mathcal{H}$ is the set of all cliques. We also show the following result, also expressible as Mutually Induced $\mathcal{H}$-packing for $\mathcal{H}$ the set of all trees or the set all stars.

▶ **Theorem 6.** *For every $\varepsilon > 0$, finding the induced (star) forest with the most edges admits a polynomial-time n^{ε}-approximation algorithms on n-vertex graphs of bounded twin-width given with an $O(1)$-sequence.*

As we already mentioned, our framework is exclusively useful for problems that are very inapproximable in general graphs; at least for which an n^{ε}-approximation algorithm is not known for every $\varepsilon > 0$. Are there natural such problems that cannot be approximated better in graphs of bounded twin-width? We answer this question positively with the example of Min Independent Dominating Set.

▶ **Theorem 7.** *For every $\varepsilon > 0$, Min Independent Dominating Set does not admit an $n^{1-\varepsilon}$-approximation algorithm in n-vertex graphs given with an $O(1)$-sequence, unless P=NP.*

The reduction is the same as the one for general graphs [18], but performed from a planar variant of 3-SAT. The obtained instances are not planar but can be contracted to planar trigraphs, hence overall have bounded twin-width.

Finally the case of Longest Path and Longest Induced Path is interesting. The best approximation factor for the former [14] is worse than $n^{0.99}$, while the latter is known to have the same inapproximability as MIS [24]. However an n^ε-approximation algorithm (for every $\varepsilon > 0$) is not excluded for Longest Path. We show that the property of bounded twin-width is unlikely to help for these two problems, as it would lead to better approximation algorithms for Longest Path in general graphs. This is mainly because subdividing at least $2 \log n$ times every edge of any n-vertex graph gives a graph with twin-width at most 4 [2].

▶ **Theorem 8.** *For any $r = \omega(1)$, an r-approximation for Longest Induced Path or Longest Path on graphs given with an $O(1)$-sequence would imply a $(1 + o(1))r$-approximation for Longest Path in general graphs.*

In turn, this can be used to exhibit a family $\mathcal{H}$ with an infinite antichain for the *induced subgraph* relation such that Mutually Induced $\mathcal{H}$-packing is *hard* to n^ε-approximate on graphs of bounded twin-width. The family $\mathcal{H}$ is simply the set of all paths terminated by triangles at both ends.

▶ **Theorem 9.** *There is an infinite family $\mathcal{H}$ of connected graphs such that if for every $\varepsilon > 0$, Mutually Induced $\mathcal{H}$-packing admits an n^ε-approximation algorithm on n-vertex graphs given with an $O(1)$-sequence, then so does Longest Path on general graphs.*

Table 1 summarizes our results and hints at future work.

Table 1 Approximability status of graph problems in general graphs and in graphs of bounded twin-width given with an $O(1)$-sequence. Everywhere "ε" should be read as "$\forall \varepsilon > 0$". Our results are enclosed by boxes. "Longest Path-hard" means that getting an r-approximation would yield essentially the same ratio for Longest Path in general graphs. The other lower bounds are under standard complexity-theoretic assumptions, mostly P≠NP. Not to clutter the table, we do not put the references, which can all be found in the paper.

Problem name	lower bound general graphs	upper bound bounded tww	lower bound bounded tww
Max Independent Set	$n^{1-\varepsilon}$	$\boxed{n^\varepsilon}$	?, self-improvement
Coloring	$n^{1-\varepsilon}$	$\boxed{n^\varepsilon}$	$4/3 - \varepsilon$
Max Induced Matching	$n^{1-\varepsilon}$	$\boxed{n^\varepsilon}$	?
Mut. Ind. H-Packing	$n^{1-\varepsilon}$	$\boxed{n^\varepsilon \text{ } (H \text{ connected})}$	?
Mut. Ind. $\mathcal{H}$-Packing	$n^{1-\varepsilon}$	$\boxed{n^\varepsilon \text{ for some } \mathcal{H}}$	$\boxed{\text{Longest Path-hard}}$
Min Ind. Dom. Set	$n^{1-\varepsilon}$	$n/\mathsf{polylog}(n)$	$\boxed{n^{1-\varepsilon}}$
Longest Path	$2^{\log^{1-\varepsilon} n}$	$n/\exp(\Omega(\sqrt{\log n}))$	$\boxed{\text{Longest Path-hard}}$
Longest Induced Path	$n^{1-\varepsilon}$	$n/\mathsf{polylog}(n)$	$\boxed{\text{Longest Path-hard}}$
Min Dominating Set	$(1-\varepsilon)\ln n$	$O(1)$	?

For the main highly inapproximable graph problems, we either obtain an n^ε-approximation algorithm on graphs of bounded twin-width given with an $O(1)$-sequence, or a conditional obstruction to such an algorithm. In the former case, can we improve further the approximation factor? The next theorem was observed using the self-improvement reduction of Feige et al. [13], which preserves the twin-width bound. This reduction consists of going from a graph G to the lexicographic product $G[G]$, where every vertex of G is replaced by a module inducing a copy of G (and iterating this trick).

▶ **Theorem 10** ([5]). *Let $r : \mathbb{N} \to \mathbb{R}$ be any non-decreasing function such that for every $\varepsilon > 0$, $r(n) = o(n^\varepsilon)$. If Max Independent Set admits an $r(n)$-approximation algorithm on n-vertex graphs of bounded twin-width given with an $O(1)$-sequence, then it further admits an $r(n)^\varepsilon$-approximation.*

To our knowledge, the application of the self-improvement trick is always to strengthen a lower bound, and never to effortlessly obtain a better approximation factor. Therefore, we may take Theorem 10 as a weak indication that our approximation ratio is best possible. Still, not even a polynomial-time approximation scheme (PTAS) is ruled out for MIS (nor for Max Induced Matching, Min Dominating Set, etc.) and we would like to see better approximation algorithms. For Coloring, as was previously observed [5], a PTAS is ruled out by the NP-hardness of deciding if a planar graph is 3-colorable or 4-chromatic, since planar graphs have twin-width at most 9 and a 9-sequence can be found in linear time [20].

Due to space restrictions, only Theorems 1–3 and half of Theorem 4 are presented in the short version of the paper. All other results as well as some deferred proofs, marked with a $\star$, can be found in the long version, in appendix.

2 Preliminaries

For i and j two integers, we denote by $[i, j]$ the set of integers that are at least i and at most j. For every integer i, $[i]$ is a shorthand for $[1, i]$.

2.1 The contraction and partition viewpoints of twin-width

A *trigraph* G has vertex set $V(G)$, black edge set $E(G)$, red edge set $R(G)$ such that $E(G) \cap R(G) = \emptyset$ (and $E(G), R(G) \subseteq \binom{V(G)}{2}$). A *contraction* in a trigraph G replaces a pair of (non-necessarily adjacent) vertices $u, v \in V(G)$ by one vertex w that is linked to $G - \{u, v\}$ in the following way to form a new trigraph G'. For every $z \in V(G) \setminus \{u, v\}$, $wz \in E(G')$ whenever $uz, vz \in E(G)$, $wz \notin E(G') \cup R(G')$ whenever $uz, vz \notin E(G) \cup R(G)$, and $wz \in R(G')$, otherwise. The *red graph* $(V(G), R(G))$ will be denoted by $\mathcal{R}(G)$. We denote by $\mathcal{T}(G)$ the *total graph* of G defined as $(V(G), E(G) \cup R(G))$. An *induced subtrigraph* of a trigraph G is obtained by removing vertices (but no edges) to G, analogously to induced subgraphs. A partial contraction sequence of an n-vertex (tri)graph G (to a trigraph H) is a sequence of trigraphs $G = G_n, \cdots, G_t = H$ for some $t \in [n]$ such that G_i is obtained from G_{i+1} by performing one contraction. A (complete) contraction sequence is such that $t = 1$, that is, H is the 1-vertex trigraph. A *d-sequence* $\mathcal{S}$ of G is a contraction sequence of G in which the red graph of every trigraph of $\mathcal{S}$ has maximum degree at most d.

Assume that there is a partial contraction sequence from a (tri)graph G to a trigraph H. If u is a vertex of H, then $u(G) \subseteq V(G)$ denotes the set of vertices eventually contracted into u in H. We denote by $\mathcal{P}(H)$ the partition $\{u(G) : u \in V(H)\}$ of $V(G)$. If G is clear from the context, we may refer to a *part* of H as any set in $\{u(G) : u \in V(H)\}$. We will mostly see d-sequences as sequences of partitions, that is, $\mathcal{P}_n, \ldots, \mathcal{P}_t$ with $\mathcal{P}_i := \{u(G) : u \in V(G_i)\}$ when $G_n, \ldots, G_t$ is a partial (contraction) d-sequence.

Given a graph G and a partition $\mathcal{P}$ of $V(G)$, the *quotient graph* of G with respect to $\mathcal{P}$ is the graph with vertex set $\mathcal{P}$, where PP' is an edge if there is $u \in P$ and $v \in P'$ such that $uv \in E(G)$. Given a (tri)graph G and a partition $\mathcal{P}$ of $V(G)$, the *quotient trigraph* $G/\mathcal{P}$ is the trigraph with vertex set $\mathcal{P}$, where PP' is a black edge if these two parts are fully adjacent – for every $u \in P$ and every $v \in P'$, $uv \in E(G)$ –, and a red edge if either there is $u \in P$ and $v \in P'$ such that $uv \in R(G)$, or there is $u_1, u_2 \in P$ and $v_1, v_2 \in P'$ such that $u_1v_1 \in E(G)$ and $u_2v_2 \notin E(G)$.

A trigraph H is a *cleanup* of another trigraph G if $V(H) = V(G)$, $R(H) \subseteq R(G)$, and $E(G) \subseteq E(H) \subseteq E(G) \cup R(G)$. That is, H is obtained from G by turning some of its red edges into black edges or non-edges. We further say that H is *full cleanup* of G if H has no red edge, and thus, is considered as a graph. Note that the total graph $\mathcal{T}(G)$ and the *black graph* $(V(G), E(G))$ of a trigraph G are extreme examples of full cleanups of G.

2.2 Balanced partition sequences

It was shown that twin-width and *versatile* twin-width (we will not need a definition here; see long version) are functionally equivalent [4]. The relevant consequence for our purposes is that every graph G with a d'-sequence admits a *balanced* d-sequence, where $d = h(d')$ depends only on d', i.e., one for which the partitions $\mathcal{P}_n, \dots, \mathcal{P}_1$ are such that for every $i \in [n]$ and $P \in \mathcal{P}_i$, $|P| \leqslant d \cdot \frac{n}{i}$. As we will resort to recursion on induced subtrigraphs and quotient trigraphs, we need to keep more information on those subinstances that the mere fact that they have twin-width at most d (otherwise the twin-width bound could quickly diverge).

This will be done by opening up the proof in [4], and handling divided $0, 1, r$-matrices with some specific properties. Thus we need to recall the relevant definitions.

Given two partitions $\mathcal{P}, \mathcal{P}'$ of the same set, we say that $\mathcal{P}'$ is a *coarsening* of $\mathcal{P}$ if every part of $\mathcal{P}$ is contained in a part of $\mathcal{P}'$, and $\mathcal{P}, \mathcal{P}'$ are distinct. Given a matrix M, we call *row division* (resp. *column division*) a partition of the rows (resp. columns) of M into parts of consecutive rows (resp. columns). A (k, ℓ)*-division*, or simply *division*, of a matrix M is a pair $(\mathcal{R} = \{R_1, \dots, R_k\}, \mathcal{C} = \{C_1, \dots, C_\ell\})$ where $\mathcal{R}$ is a row division and $\mathcal{C}$ is a column division. In a matrix division $(\mathcal{R}, \mathcal{C})$, each part $R \in \mathcal{R}$ is called a *row part*, and each part $C \in \mathcal{C}$ is called a *column part*. Given a subset R of rows and a subset C of columns in a matrix M, the *zone* $M[R, C]$ denotes the submatrix of all entries of M at the intersection between a row of R and a column of C. A *zone* of a matrix partitioned by $(\mathcal{R}, \mathcal{C}) = (\{R_1, \dots, R_k\}, \{C_1, \dots, C_\ell\})$ is any $M[R_i, C_j]$ for $i \in [k]$ and $j \in [\ell]$. A zone is *constant* if all its entries are identical, *horizontal* if all its columns are equal, and *vertical* if all its rows are equal. A *0,1-corner* is a 2×2 $0, 1$-matrix which is neither horizontal nor vertical.

Unsurprisingly, $0, 1, r$*-matrices* are such that each entry is in $\{0, 1, r\}$ where r is an error symbol that should be understood as a red edge. A *neat division* of a $0, 1, r$-matrix is a division for which every zone either contains only r entries or contains no r entry and is horizontal or vertical (or both, i.e., constant). Zones filled with r entries are called *mixed*. A *neatly divided matrix* is a pair $(M, (\mathcal{R}, \mathcal{C}))$ where M is a $0, 1, r$-matrix and $(\mathcal{R}, \mathcal{C})$ is a neat division of M. A t*-mixed minor* in a neatly divided matrix is a (t, t)-division which coarsens the neat subdivision, and contains in each of its t^2 zones at least one mixed zone (i.e., filled with r entries) or a 0,1-corner. A neatly divided matrix is said t*-mixed free* if it does not admit a t-mixed minor.

A *mixed cut of a row part* $R \in \mathcal{R}$ of a neatly divided matrix $(M, (\mathcal{R}, \mathcal{C} = \{C_1, C_2, \dots\}))$ is an index i such that both $M[R, C_i]$ and $M[R, C_{i+1}]$ are not mixed, and there is a $0, 1$-corner in the 2-by-$|R|$ zone defined by the last column of C_i, the first column of C_{i+1}, and R. The *mixed value of a row part* $R \in \mathcal{R}$ of a neatly divided matrix $(M, (\mathcal{R}, \mathcal{C} = \{C_1, C_2, \dots\}))$ is the number of mixed zones $M[R, C_j]$ plus the number of mixed cuts of R. We similarly define the mixed value of a column part $C \in \mathcal{C}$. The *mixed value of a neat division* of a $0, 1, r$-matrix is the maximum of the mixed values taken over every part. The *part size* of a division $(\mathcal{R}, \mathcal{C})$ is defined as $\max(\max_{R \in \mathcal{R}} |R|, \max_{C \in \mathcal{C}} |C|)$. A division is *symmetric* if the largest row index of each row part and the largest column index of each column part define the same set of integers. We call *symmetric fusion* of a symmetric division the contraction of two consecutive parts in $\mathcal{C}$ and of the two corresponding parts in $\mathcal{R}$. A symmetric fusion on a symmetric division yields another symmetric division. A matrix $A := (a_{i,j})_{i,j}$ is said *symmetric* in the usual sense, namely, for every entry $a_{i,j}$ of A, $a_{i,j} = a_{j,i}$.

In what follows, we set $c_d := 8/3(d+1)^2 2^{4d}$.

▶ **Definition 11.** *Let $\mathcal{M}_{n,d}$ be the class of the neatly divided $n \times n$ symmetric $0,1,r$-matrices $(M,(\mathcal{R},\mathcal{C}))$, such that $(\mathcal{R},\mathcal{C})$ is symmetric and has:*
- *mixed value at most $4c_d$,*
- *part size at most 2^{4c_d+2}, and*
- *no d-mixed minor.*

We show in the long version the next couple of key lemmas.

▶ **Lemma 12** (⋆). *Let d be a natural, $s := 2^{4c_d+4}$, and $d' := c_d \cdot 2^{4c_d+4}$. Given an n-vertex graph G with a d-sequence, one can compute in time $n^{O(1)}$ a partition $\mathcal{P} = \{P_1, P_2, \ldots, P_{\lfloor\sqrt{n}\rfloor}\}$ of $V(G)$ satisfying*
- *for every integer $1 \leqslant i \leqslant \lfloor\sqrt{n}\rfloor$, $|P_i| \leqslant s\sqrt{n} \leqslant d'\sqrt{n}$, and*
- *the red graph of $G/\mathcal{P}$ has maximum degree at most d'.*

A neatly divided matrix $(M,(\mathcal{R},\mathcal{C}))$ is said *conform* to a trigraph G if M is the adjacency matrix of a trigraph G' such that G is a cleanup of G'.

▶ **Lemma 13** (⋆). *Let $\hat{d}$ be a natural, $d = 2\hat{d}+2$, and set $s := 2^{4c_d+4}$, and $d' := c_d \cdot 2^{4c_d+4}$. Given an n-vertex graph G with a $\hat{d}$-sequence, or an n-vertex trigraph G with a neatly divided matrix $(M,(\mathcal{R},\mathcal{C})) \in \mathcal{M}_{n,d}$ such that M is conform to G, one can compute in time $n^{O(1)}$ a partition $\mathcal{P} = \{P_1, P_2, \ldots, P_{\lfloor\sqrt{n}\rfloor}\}$ of $V(G)$ with maximum red degree at most d' satisfying that, for every integer $1 \leqslant i \leqslant \lfloor\sqrt{n}\rfloor$, $|P_i| \leqslant s\sqrt{n} \leqslant d'\sqrt{n}$, and for any trigraph H that is*
- *a cleanup of an induced subtrigraph of $G/\mathcal{P}$, or*
- *an induced subtrigraph $G[\bigcup_{i\in J\subseteq[\lfloor\sqrt{n}\rfloor]} P_i]$,*

a neatly divided matrix $(M',(\mathcal{R}',\mathcal{C}')) \in \mathcal{M}_{|V(H)|,d}$ conform to H can be computed in time $n^{O(1)}$.

3 Approximation algorithms for Max Independent Set

We naturally start our study with Max Independent Set, a central problem that is very inapproximable [19, 27], and yet constitutes the textbook example of our approach.

3.1 Subexponential-time constant-approximation algorithm

We present a subexponential-time $O_d(1)$-approximation for WMIS on graphs given with a d-sequence, which we recall, is unlikely to exist in general graphs [10].

▶ **Lemma 14.** *Let d' be a natural, $s := 2^{4c_{d'}+4}$, and $d := c_{d'} \cdot 2^{4c_{d'}+4}$. Assume n-vertex inputs G, vertex-weighted by w, are given with a d'-sequence. Weighted Max Independent Set can be $(d+1)$-approximated in time $2^{O_d(\sqrt{n})}$ on these inputs.*

Proof. By Lemma 12, we compute in polynomial time a partition $\mathcal{P} = \{P_1, \ldots, P_{\lfloor\sqrt{n}\rfloor}\}$ of $V(G)$ whose parts have size at most $s\sqrt{n}$ and such that $\mathcal{R}(G/\mathcal{P})$ has maximum degree at most d.

For every integer $1 \leqslant i \leqslant \lfloor\sqrt{n}\rfloor$, we compute a heaviest independent set in $G[P_i]$, say S_i. Even with an exhaustive algorithm, this takes time $\sqrt{n} \cdot s^2 n \cdot 2^{s\sqrt{n}} = 2^{O_d(\sqrt{n})}$. We then $(d+1)$-color (in linear time) $\mathcal{R}(G/\mathcal{P})$, which is possible since this graph has maximum degree at most d. This defines a coarsening of $\mathcal{P}$ in $d+1$ parts $\mathcal{Q} = \{C_1, \ldots, C_{d+1}\}$. Thus, $\mathcal{Q}$ is a partition of $V(G)$ such that C_j consists of all the parts $P_i \in \mathcal{P}$ receiving color j in the $(d+1)$-coloring of $\mathcal{R}(G/\mathcal{P})$.

For every $j \in [d+1]$, let H_j be the *graph* $(G/\mathcal{P})[C_j]$[4] vertex-weighted by $P_i \subseteq C_j \mapsto w(S_i)$. Note that $(G/\mathcal{P})[C_j]$ can indeed be assimilated to a graph, since it has, by design, no red edge. We compute a heaviest independent set in H_j, say R_j. This takes time $(d+1)\cdot n \cdot 2^{\sqrt{n}} = 2^{O_d(\sqrt{n})}$. We output $\bigcup_{P_i \subseteq R_j} S_i$ for the index $j \in [d+1]$ maximizing $\sum_{P_i \subseteq R_j} w(S_i)$.

This finishes the description of the algorithm. We already argued that its running time is $2^{O_d(\sqrt{n})}$. We shall justify that it does output an independent set of weight at least a $\frac{1}{d+1}$ fraction of the optimum $\alpha(G)$. Let I be the output of the algorithm.

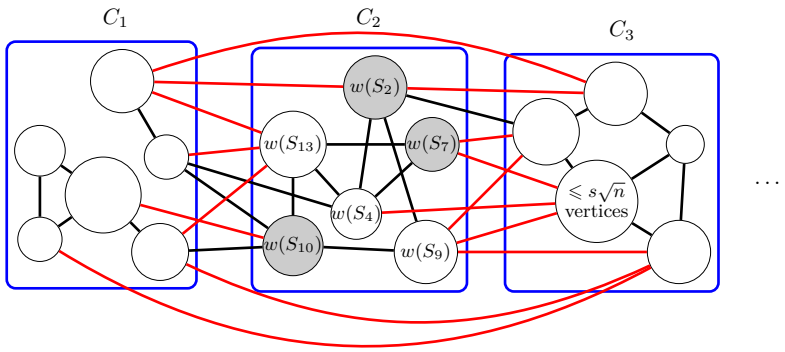

Figure 2 The trigraph $G/\mathcal{P}$ with its $\lfloor\sqrt{n}\rfloor$ vertices, each corresponding to a subset of at most $s\sqrt{n}$ vertices of G. The weights $w(S_i)$ of heaviest independent sets S_i of $G[P_i]$ for each part P_i of the color class C_2 of the $d+1$-coloring of $\mathcal{R}(G/\mathcal{P})$. A heaviest independent set in the so-weighted $(G/\mathcal{P})[C_2]$ (shaded) corresponds to an optimum solution in $G[\bigcup_{P_i \subseteq C_2} P_i]$. One of these $d+1$ independent sets is a $d+1$-approximation.

***I* is indeed an independent set.** For any $j \in [d+1]$, consider two vertices $x, y \in \bigcup_{P_i \subseteq R_j} S_i$. If $\{x, y\} \in S_i$ for some i, then x and y are non-adjacent since S_i is an independent set of $G[P_i]$. Else $x \in S_i$ and $y \in S_{i'}$ for some $i \neq i'$. P_i and $P_{i'}$ are not linked by a black edge in $(G/\mathcal{P})[C_j]$ since R_j is an independent set in H_j, nor they can be linked by a red edge (there are none in $(G/\mathcal{P})[C_j]$). Thus again, x and y are non-adjacent in G.

***I* has weight at least $\frac{\alpha(G)}{d+1}$.** We claim that $\bigcup_{P_i \subseteq R_j} S_i$ is a heaviest independent set of $G[C_j]$. Note that the P_is that are included in C_j (and partition it) form a module partition of $G[C_j]$. In particular, any heaviest independent set intersecting some $P_i \subseteq C_j$ has to contain a heaviest independent of $G[P_i]$. This is precisely what the algorithm computes. Then a heaviest independent set in $G[C_j]$ packs such subsolutions to maximize the total weight, which is what is computed in H_j.

We conclude by the pigeonhole principle, since a heaviest independent set X of G is such that $w(X \cap C_j) \geqslant \frac{\alpha(G)}{d+1}$ for some $j \in [d+1]$. ◀

3.2 Time-approximation trade-offs

Lemma 14 runs exhaustive algorithms on induced subgraphs of size $O_d(\sqrt{n})$. As such, the latter inputs keep the same twin-width upper bound. To speed up the algorithm (admittedly while worsening the approximation factor) it is tempting to recursively call our very algorithm. We show that this leads to a time-approximation trade-off parameterized

[4] We use this notation as a slight abuse of notation for $(G/\mathcal{P})[\{P_i \ : \ P_i \subseteq C_j\}]$.

by an integer $q = 0, \ldots, O_d(\log\log n)$. At one end of this discrete curve, one finds the exact exponential algorithm ($q = 0$), and more interestingly the $d+1$-approximation in time $2^{O_d(\sqrt{n})}$ ($q = 1$), while at the other end lies a polynomial-time algorithm with approximation factor n^ε, where $\varepsilon > 0$ can be made as small as desired.

As we will deal with the same kind of recursions for several problems, we show the following generic abstraction.

▶ **Lemma 15.** *Let $\hat{d}$ be a natural, $d' = 2\hat{d}+2$, and $d := c_{d'} \cdot 2^{4c_{d'}+4}$. Let Π be an optimization graph problem where inputs come with a $\hat{d}$-sequence of their n-vertex graph G, or with a neatly divided matrix $(M, (\mathcal{R}, \mathcal{C})) \in \mathcal{M}_{n,d'}$ conform to G. Let $\mathcal{P}$ be the partition of $V(G)$ given by Lemma 13. Assume that*

1. *Π can be exactly solved in time $2^{O(n)}$, and there are constants c_1, c_2, c_3, and a function $f \geqslant 1$ such that*
2. *a $d^{c_3} r^2$-approximation of Π on G can be built in time n^{c_2} by using at most n^{c_1} calls to an r-approximation of Π –or another optimization problem Π' already satisfying the conclusion of the lemma– on an induced subgraph of G with at most $f(d)\sqrt{n}$ vertices or a full cleanup of an induced subtrigraph of $G/\mathcal{P}$ (on at most $\sqrt{n}$ vertices).*

Then Π can be $d^{c_3(2^q-1)}$-approximated in time

$$(f(d)^q n)^{(2-2^{-q})(c_1+c_2)} \cdot 2^{f(d)^{2(1-2^{-q})} n^{2^{-q}}},$$

for any non-negative integer q.

Proof. The proof is by induction on q. The case $q = 0$ is implied by Item 1. The case $q = 1$, and the induction step in general, is nothing more than an abstraction of Lemma 14, where exhaustive algorithms are replaced by recursive calls.

For any $q \geqslant 0$, we assume that Π can $d^{c_3(2^q-1)}$-approximated in the claimed running time, and show the same statement for the value $q+1$. Following Item 2, we run this algorithm –or one for another optimization problem Π' satisfying the conclusion of the lemma– at most n^{c_1} times on $f(d)\sqrt{n}$-vertex induced subgraphs of the input graph G or on full cleanups of induced subtrigraphs of $G/\mathcal{P}$. The latter graphs have at most $\sqrt{n} \leqslant f(d)\sqrt{n}$ vertices. By Lemma 13, we can compute in polynomial time a neatly divided matrix $(M', (\mathcal{R}', \mathcal{C}')) \in \mathcal{M}_{|V(H)|,d'}$ conform to H, for each graph H of a recursive call; hence the induction applies.

Overall this takes time at most

$$n^{c_1} + n^{c_2} \cdot \left((f(d)^q \cdot f(d)\sqrt{n})^{(2-2^{-q})(c_1+c_2)} \cdot 2^{f(d)^{2(1-2^{-q})} (f(d)\sqrt{n})^{2^{-q}}} \right)$$

$$\leqslant (f(d)^{q+1} n)^{c_1+c_2+\frac{1}{2}(2-2^{-q})(c_1+c_2)} \cdot 2^{f(d)^{2(1-2^{-q})+2^{-q}} n^{\frac{2^{-q}}{2}}}$$

$$= (f(d)^{q+1} n)^{(2-\frac{2^{-q}}{2})(c_1+c_2)} \cdot 2^{f(d)^{2-2^{-q+1}+2^{-q}} n^{2^{-(q+1)}}}$$

$$= (f(d)^{q+1} n)^{(2-2^{-(q+1)})(c_1+c_2)} \cdot 2^{f(d)^{2(1-2^{-(q+1)})} n^{2^{-(q+1)}}}.$$

For the first inequality, we assume that the two summands are larger than 2, so their sum can be bounded by their product.

Besides we get an approximation of factor at most $(d^{c_3(2^q-1)})^2 d^{c_3} = d^{c_3(2^{q+1}-1)}$. ◀

In more legible terms we have proved that:

▶ **Lemma 16.** *Problems Π satisfying the assumptions of Lemma 15 can be $d^{O(1)(2^q-1)}$-approximated in time $2^{O_{d,q}(\sqrt[2^q]{n})}$, for any non-negative integer q.*

While most graph problems admit single-exponential algorithms, we will deal with such a problem that is only known to be solvable in time $2^{O(n \log n)}$. Therefore we prove a variant of Lemma 15 with a slightly worse running time.

▶ **Lemma 17** (⋆). *Let Π be solvable in time $2^{O(n \log n)}$ and satisfy the second item of Lemma 15. Then Π can be $d^{c_3(2^q-1)}$-approximated in time*

$$2^{\left((c_1+c_2)(2-2^{-q})\log f(d)+f(d)^{2(1-2^{-q})}n^{2^{-q}}\right)\log n},$$

for any non-negative integer q.

Again the previous lemma can be rewritten as:

▶ **Lemma 18.** *Problems Π satisfying the assumptions of Lemma 17 can be $d^{O(1)(2^q-1))}$-approximated in time $2^{O_{d,q}(\sqrt[2^q]{n}\log n)}$, for any non-negative integer q.*

We derive from Lemma 17 the following notable regimes.

▶ **Theorem 19** (⋆). *Problems Π satisfying the assumptions of Lemma 17 admit polynomial-time n^{ε}-approximation algorithms, for any $\varepsilon > 0$.*

▶ **Theorem 20** (⋆). *Problems Π satisfying the assumptions of Lemma 15, resp. Lemma 17, admit a $\log n$-approximation algorithm running in time $2^{O_d(n^{\frac{1}{\log\log n}})}$, resp. $2^{O_d(n^{\frac{1}{\log\log n}}\log n)}$.*

We derive the following for Weighted Max Independent Set.

▶ **Theorem 21** (⋆). *Weighted Max Independent Set on n-vertex graphs G (vertex-weighted by w) given with a d'-sequence satisfies the assumptions of Lemma 15. In particular, this problem admits*

- *a $(d+1)^{2^q-1}$-approximation in time $2^{O_{d,q}(n^{2^{-q}})}$, for every integer $q \geqslant 0$,*
- *an n^{ε}-approximation in polynomial-time $O_{d,\varepsilon}(1)\log^{O_d(1)} n \cdot n^{O(1)}$, for any $\varepsilon > 0$, and*
- *a $\log n$-approximation in time $2^{O_d(n^{\frac{1}{\log\log n}})}$,*

with $d := c_{2d'+2} \cdot 2^{4c_{2d'+2}+4}$.

4 Finding the suitable generalization: the case of Coloring

In this section, we deal with the Coloring problem. Unlike for WMIS, we cannot solely resort to recursively calling our Coloring algorithm on smaller graphs. The right problem generalization needs to be found for the inductive calls to work through, and it happens to be Set Coloring.

In the Set Coloring problem, the input is a couple (G, b) where G is a graph, and b is a function assigning a positive integer to each vertex of G. The goal is to find, for each $v \in V(G)$, a set S_v of at least $b(v)$ colors such that $S_u \cap S_v = \emptyset$ whenever $uv \in E(G)$, and minimizing $|\cup_{v\in V(G)} S_v|$. Let $\chi_b(G)$ be the optimal value of Set Coloring for (G, b). Observe that Coloring corresponds to the case where $b(v) = 1$ for every $v \in V(G)$.

▶ **Theorem 22.** *Set Coloring (and hence Coloring) on n-vertex graphs G given with a d'-sequence satisfies the assumptions of Lemma 17. In particular, this problem admits*

- *a $(d+1)^{2^q-1}$-approximation in time $2^{O_{d,q}(n^{2^{-q}}\log n)}$, for every integer $q \geqslant 0$, and*
- *an n^{ε}-approximation in polynomial-time for any $\varepsilon > 0$.*

with $d := c_{2d'+2} \cdot 2^{4c_{2d'+2}+4}$.

Proof. It is known [25] that Set Coloring can be solved using the inclusion-exclusion principle in time $O^*(\max_{v\in V(G)} b(v)^n) = 2^{O(n\log n)}$. We now prove that it satisfies the second item of Lemma 15. We denote by $\mathcal{A}$ the r-approximation algorithm of the statement, which we will use on instances of Set Coloring. In particular, we will call it at most $\sqrt{n}+1$ times, and will obtain at the end a $(d+1)r^2$-approximation on our input (G,b) in polynomial time.

We first apply Lemma 13 to get, in polynomial-time, a partition $\mathcal{P} = \{P_1,\dots,P_{\lfloor\sqrt{n}\rfloor}\}$ of $V(G)$ whose parts have size at most $d\sqrt{n}$ and such that $R(G/\mathcal{P})$ has maximum degree at most d. For every $i \in [\lfloor\sqrt{n}\rfloor]$, we use $\mathcal{A}$ to compute an r-approximated solution c_{P_i} of $(G[P_i], b_{|P_i})$. We denote by b' the function which assigns, to each P_i, the number of colors of c_{P_i}. We now compute, in polynomial-time, a proper $(d+1)$-coloring of $R(G/\mathcal{P})$, which defines the sets C_1, …, C_{d+1}. For each $j \in [d+1]$, we construct another Set Coloring instance consisting of the graph $H_j = (G/\mathcal{P})[C_j]$ (recall that this trigraph has no red edge, and can thus be seen as a graph), together with the function $b'_{|C_j}$. Again we use $\mathcal{A}$ to compute an r-approximated solution on $(H_j, b'_{|C_j})$. We denote by c_H this solution. Let G_j be the subgraph of G induced by $\cup_{P_i\in C_j} P_i$, and b_j the restriction of b to $V(G_j)$. We now show how to construct a solution c_j of Set Coloring to (G_j, b_j) from c_H and all c_{P_i}. Recall that for every $P_i \in C_j$, every $v \in P_i$, we have that $c_{P_i}(v)$ is a subset of $\{1,\dots,b'(P_i)\}$ of size at least $b(v)$, and that $c_H(P_i)$ is a subset of size at least $b'(P_i)$. Hence, for each $P_i \in C_j$, one can choose an arbitrary bijection τ from $\{1,\dots,b'(P_i)\}$ to $c_H(P_i)$, and define to each vertex $v \in P_i$ the set $c_j(v)$ as $\{\tau(x) : x \in c_{P_i}(v)\}$.

By construction, this solution is a feasible one for the instance (G_j, b_j). Let us prove that it is an r^2-approximation of $\chi_{b_j}(G_j)$. First, by definition of c_H, our solution uses at most $r\cdot\chi_{b'_{|C_j}}(H_j)$ colors. Then, by definition of c_{P_i} for every $P_i \in C_j$, we have $b'_{C_j}(P_i) \leqslant r\cdot\chi_{b_{|P_i}}(G[P_i])$. Now, denote by Γ the function which assigns to each $P_i \in C_j$ the number $\chi_{b_{|P_i}}(G[P_i])$. We now use the following claim, whose proof is left to the reader.

▷ Claim 23. Let (G,b) be an instance of Set Coloring, and $r \in \mathbb{R}_+$. It holds that $\chi_{r\cdot b}(G) \leqslant r\cdot\chi_b(G)$, where $r\cdot b$ is the function which assigns $r\cdot b(v)$ to each $v \in V(G)$.

This implies $\chi_{b'_{|C_j}}(H_j) \leqslant r\cdot\chi_\Gamma(H_j)$, and thus our solution uses at most $r^2\cdot\chi_\Gamma(H_j)$ colors. We now prove the following claim.

▷ Claim 24. $\chi_\Gamma(H_j) \leqslant \chi_{b_j}(G_j)$.

Proof of the claim. Let c be an optimal solution for (G_j, b_j). For every distinct P_i, $P_{i'} \in C_j$ such that $P_iP_{i'}$ is an edge of H_j, it holds that there are all possible edges between P_i and $P_{i'}$ in G_j (by definition of the coloring C_1, ⋯, C_{d+1}), hence it holds that $\bigcup_{v\in P_i} c(v)$ and $\bigcup_{v\in P_{i'}} c(v)$ have empty intersection. Moreover, by definition of Γ, we have that $\bigcup_{v\in P_i} c(v)$ is of size at least $\Gamma(P_i)$, hence the function which assigns $\bigcup_{v\in P_i} c(v)$ to each P_i is a feasible solution for (H_j, Γ) using at most $\chi_{b_j}(G_j)$ colors. ◁

We now have in hand an r^2-approximated solution of (G_j, b_j) for every $j \in [d+1]$, which can be turned into a $(d+1)r^2$-approximated solution of (G,b), as desired. ◀

References

1 Nikhil Bansal, Parinya Chalermsook, Bundit Laekhanukit, Danupon Nanongkai, and Jesper Nederlof. New tools and connections for exponential-time approximation. *Algorithmica*, 81(10):3993–4009, 2019. doi:10.1007/s00453-018-0512-8.

2 Pierre Bergé, Édouard Bonnet, and Hugues Déprés. Deciding twin-width at most 4 is NP-complete. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 18:1–18:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.18.

3 Édouard Bonnet, Dibyayan Chakraborty, Eun Jung Kim, Noleen Köhler, Raul Lopes, and Stéphan Thomassé. Twin-width VIII: delineation and win-wins. *CoRR*, abs/2204.00722, 2022. doi:10.48550/arXiv.2204.00722.

4 Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width II: small classes. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1977–1996, 2021. doi:10.1137/1.9781611976465.118.

5 Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width III: max independent set, min dominating set, and coloring. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 35:1–35:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.35.

6 Édouard Bonnet, Ugo Giocanti, Patrice Ossona de Mendez, Pierre Simon, Stéphan Thomassé, and Szymon Torunczyk. Twin-width IV: ordered graphs and matrices. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20–24, 2022*, pages 924–937. ACM, 2022. doi:10.1145/3519935.3520037.

7 Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable FO model checking. *J. ACM*, 69(1):3:1–3:46, 2022. doi:10.1145/3486655.

8 Nicolas Bourgeois, Bruno Escoffier, and Vangelis Th. Paschos. Approximation of max independent set, min vertex cover and related problems by moderately exponential algorithms. *Discret. Appl. Math.*, 159(17):1954–1970, 2011. doi:10.1016/j.dam.2011.07.009.

9 Parinya Chalermsook, Bundit Laekhanukit, and Danupon Nanongkai. Graph products revisited: Tight approximation hardness of induced matching, poset dimension and more. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1557–1576. SIAM, 2013. doi:10.1137/1.9781611973105.112.

10 Parinya Chalermsook, Bundit Laekhanukit, and Danupon Nanongkai. Independent set, induced matching, and pricing: Connections and tight (subexponential time) approximation hardnesses. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 370–379, 2013. doi:10.1109/FOCS.2013.47.

11 Marek Cygan, Lukasz Kowalik, Marcin Pilipczuk, and Mateusz Wykurz. Exponential-time approximation of hard problems. *CoRR*, abs/0810.4934, 2008. arXiv:0810.4934.

12 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 624–633. ACM, 2014. doi:10.1145/2591796.2591884.

13 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Approximating clique is almost NP-complete (preliminary version). In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 2–12. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185341.

14 Harold N. Gabow and Shuxin Nie. Finding a long directed cycle. *ACM Trans. Algorithms*, 4(1):7:1–7:21, 2008. doi:10.1145/1328911.1328918.

15 Jakub Gajarský, Michal Pilipczuk, Wojciech Przybyszewski, and Szymon Torunczyk. Twin-width and types. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 123:1–123:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.123.

16 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

17 Sylvain Guillemot and Dániel Marx. Finding small patterns in permutations in linear time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 82–101, 2014. doi:10.1137/1.9781611973402.7.

18 Magnús M. Halldórsson. Approximating the minimum maximal independence number. *Inf. Process. Lett.*, 46(4):169–172, 1993. doi:10.1016/0020-0190(93)90022-2.

19 Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 627–636, 1996. doi:10.1109/SFCS.1996.548522.

20 Petr Hliněný. Twin-width of planar graphs is at most 9, 2022. doi:10.48550/arXiv.2205.05378.

21 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

22 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

23 Stefan Kratsch, Florian Nelles, and Alexandre Simon. On triangle counting parameterized by twin-width. *CoRR*, abs/2202.06708, 2022. arXiv:2202.06708.

24 Carsten Lund and Mihalis Yannakakis. The approximation of maximum subgraph problems. In Andrzej Lingas, Rolf G. Karlsson, and Svante Carlsson, editors, *Automata, Languages and Programming, 20nd International Colloquium, ICALP93, Lund, Sweden, July 5-9, 1993, Proceedings*, volume 700 of *Lecture Notes in Computer Science*, pages 40–51. Springer, 1993. doi:10.1007/3-540-56939-1_60.

25 Jesper Nederlof. Inclusion exclusion for hard problems, 2008.

26 Michal Pilipczuk, Marek Sokolowski, and Anna Zych-Pawlewicz. Compact representation for matrices of bounded twin-width. In Petra Berenbrink and Benjamin Monmege, editors, *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*, volume 219 of *LIPIcs*, pages 52:1–52:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.STACS.2022.52.

27 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007.